(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 785 498 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
07.01.1999 Bulletin 1999/01

(43) Date of publication A2:
23.07.1997 Bulletin 1997/30

(21) Application number: 97100717.4

(22) Date of filing: 17.01.1997

(51) Int. Cl.^{6}: **G06F 1/32**, G06F 11/34

(84) Designated Contracting States:
DE FR GB IT NL

(30) Priority: 17.01.1996 US 10136

(71) Applicant:
TEXAS INSTRUMENTS INCORPORATED
Dallas Texas 75265 (US)

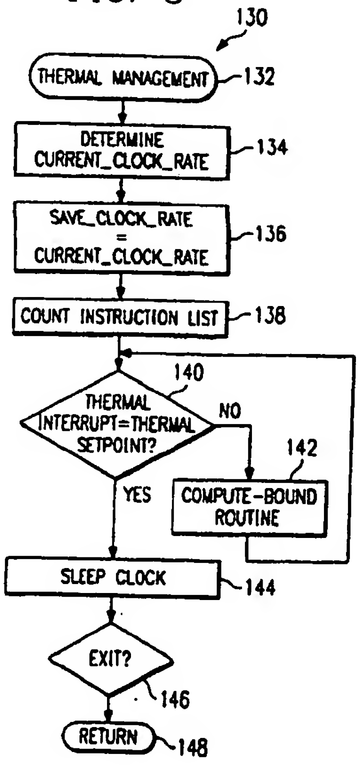
(72) Inventor:
Watts, LaVaughn F., Jr.
Temple, TX 76502 (US)

(74) Representative:
Schwepfinger, Karl-Heinz, Dipl.-Ing. et al
Prinz & Partner GbR
Manzingerweg 7
81241 München (DE)

(54) Method and system for controlling sensed dynamic operating characteristics of a CPU

(57) A method and system (130) for controlling sensed CPU dynamic operating characteristics includes the steps of and circuitry for sensing at least one dynamic CPU operating characteristic (140) while the CPU operates at a first clock rate (134). The system (130) determines that a setpoint interrupt condition exists (140) by virtue of the at least one sensed CPU dynamic operating characteristic establishing a predetermined relationship relative to a predetermined setpoint (140) that associates with the at least one dynamic operating characteristic. In the event that the setpoint interrupt condition exists, the circuitry and instructions control (144) the clock rate relative to the first clock rate. In the event that the setpoint interrupt condition does not exist, the circuitry and instructions repeat the above steps of determining the interrupt condition and controlling the clock rate. The method and system (130) also determine whether the CPU is in a compute-bound state (142). This operation in conjunction with a realtime power conservation apparatus and method (20) is a particularly attractive feature of the present invention.

FIG. 3



Printed by Xerox (UK) Business Services
2.16.5/3.4

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 97 10 0717

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	EP 0 566 395 A (DIA SEMICON SYSTEMS INC) 20 October 1993 * column 2, line 32 - column 3, line 16 * ---	1-28	G06F1/32 G06F11/34 G06F1/20
X	US 5 451 892 A (BAILEY JOSEPH A) 19 September 1995 * column 1, line 47 - column 3, line 64 * ---	1-28	
X	EP 0 683 558 A (SYMBIOS LOGIC INC) 22 November 1995 * column 1, line 25 - column 3, line 11 * ---	1-28	
A	EP 0 426 410 A (TEXAS INSTRUMENTS INC) 8 May 1991 * the whole document * -----	1-28	

TECHNICAL FIELDS SEARCHED (Int.Cl.6)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	17 November 1998	Nygren, P

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 :
G06F 9/38, 13/12, G06T 1/00

A1

(11) International Publication Number: WO 00/00887

(43) International Publication Date: 6 January 2000 (06.01.00)

(21) International Application Number: PCT/US99/14889

(22) International Filing Date: 30 June 1999 (30.06.99)

(30) Priority Data:
60/091,401 30 June 1998 (30.06.98) US

(71) Applicant: INTERGRAPH CORPORATION [US/US]; One Madison Industrial Park, Huntsville, AL 35894-0001 (US).

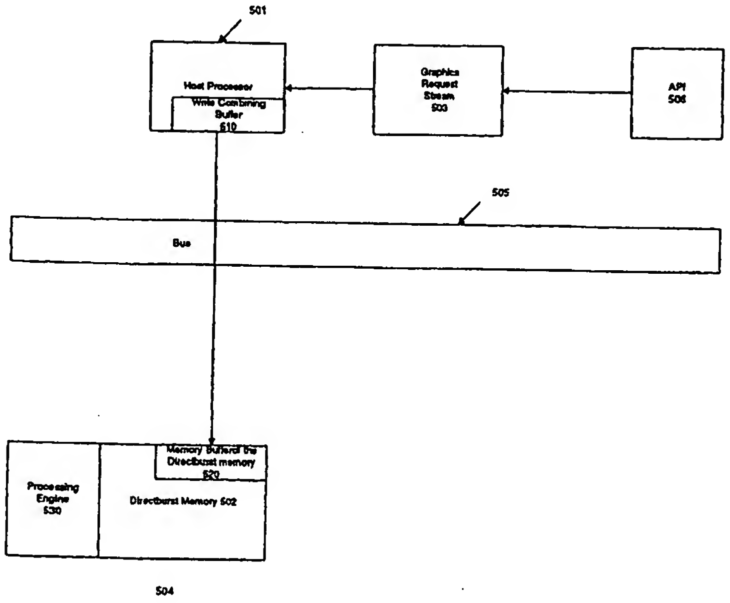
(72) Inventors: TERRY, Joseph, Clay; 3219 Highland Plaza, Huntsville, AL 35801 (US). KIRKLAND, Dale, L.; 106 Spring Water Drive, Madison, AL 35758 (US). CONKLIN, Steve; 514 Mill Road, Madison, AL 35758 (US). QUINN, Matthew, C.; 201 Water Hill Road, Madison, AL 35758 (US).

(74) Agents: SUNSTEIN, Bruce, D. et al.; Bromberg & Sunstein LLP, 125 Summer Street, Boston, MA 02110-1618 (US).

(81) Designated States: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD AND APPARATUS FOR TRANSPORTING INFORMATION TO A GRAPHIC ACCELERATOR CARD


501
Host Processor
Write Combining Buffer 510
Graphics Request Stream 503
API 506
505
Bus
Memory Buffer of the Directburst memory 520
Processing Engine 530
Directburst Memory 502
504

(57) Abstract

A graphics request stream is transferred from a host processor to a graphics card via a host bus so that the stream traverses the host bus no more than once. To that end, the graphics card has a graphics card memory, and the host processor has a host memory configured in a first memory configuration. The graphics card memory may be configured in the first memory configuration, and the graphics request stream is received directly in a message from the host processor (via the host bus). Upon receipt by the graphics card, the graphics request stream is written to the graphics card memory.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

Method and Apparatus for Transporting Information to a Graphic Accelerator Card

Field of the Invention

The present invention is related to graphics accelerator cards and, more particularly, involves the use of memory on graphics accelerator cards.

Background of the Invention

Typical computer systems employ a graphics accelerator card for enhancing the resolution and the display of graphics. The display of graphics requires a two part process, rendering and geometry acceleration. In prior art graphics cards, the geometry phase was performed by the central processing unit (CPU) of the computer system while the rendering phase was performed by the graphics card. The (CPU) is often referred to as a host processor. This often overloaded the CPU, since graphics were vying for processor time with external applications. Currently, high-end graphics cards have been configured to perform both the rendering phase and the geometry phase. This system improves performance and graphic rendering because the central processing unit is free to perform other processes while the graphics are being processed on the graphics card.

Although performance is increased during processing by having the graphics card perform both rendering and geometry acceleration, the graphics request must still be sent to the graphics card through the CPU which involves significant memory swaps between RAM memory and cache memory associated with the CPU.

See Fig.1 for a schematic diagram of the components involved in an exemplary prior art graphics card. Fig. 1 shows a host processor **9** of a computer system which is connected to a bus **1**. The bus **1** is used for transporting information to and from various components of the computer system, including main memory **7**. The host processor **9** receives a request from an application level program to create a graphics display. The request may be in the form of a group of instructions which accesses an application program interface ("API") **11**. The

API converts the instructions into a graphics request stream **10** which is capable of being understood by the graphics accelerator. The graphics request stream **10** is transmitted to a cache 8 associated with the host processor, and placed into a cache line via bus **1**. The graphics request stream is transported from the cache **8** across the bus **1** and deposited in a graphics memory location **106** of the graphics card **104**. The graphics request stream **10** is processed by a graphics processor **105** and then sent to a display device.

Figure 2 shows a prior art method of receiving the graphics request and transporting the graphics request stream to the graphics accelerator card for processing. The process begins at step **302**, in which an application level program makes a request for a graphics display. This causes the appropriate functions of the API **11** to be called. The result of the API functions form a graphics request stream **10** based on the request from the application level program in step **304**.

The host processor **9** writes the graphics request stream **10** to main memory **7** in step **306**, which requires the graphics request stream to pass across the system bus. Cache read and write is indicated by a subscript numeral in Fig. 1. Because the position in main memory 7 that is written to is typically not in the cache **8**, and the cache line usually has data in it that is not coherent with main memory **7**, a cache line swap must take place. This involves writing the current cache line contents into an associated main memory location 7, (step **308**), and writing the newly addressed cache line **12** having the graphics request stream into the cache (step **310**). Thus, writing the graphics request stream to the cache of the CPU requires the graphics request stream to pass across the system bus twice. Once the data of the graphics request stream **10** is cached in the cache memory, it still must be moved into the graphics system before rendering can occur, thus requiring a third crossing of the system bus, (step **312**). To do this, a graphics processor **105** on the graphics card **104** is controlled by driver software. The driver software causes the host processor to read the graphics request stream **10** from the cached memory **8**, and then passes the graphics request stream to the graphics processor **105** of the graphics card which writes it into a memory location **106** for processing (step **314**). Once initiated, the graphics processor **105** proceeds without further intervention by the CPU **9**, and the processed graphics request stream is displayed by a display device, (step **316**).

In summary, each word of data of the graphics request stream that is moved into the graphics accelerator requires two transactions for storage in cache memory, and one transaction to move it from cache memory 8 to the graphics pipeline **106**. Processing data in this way thus requires at least three read/writes across the system bus, consequently reducing the rendering speed to no faster than about thirty-three percent of the system bus rate.

Summary of the Invention

In accordance with one aspect of the invention, a graphics request stream is transferred from a host processor to a graphics card via a host bus so that the stream traverses the bus no more than once. To that end, the graphics card has a graphics card memory, and the host processor has an address system for addressing the graphics card memory. In accordance with preferred embodiments of the invention, the graphics card receives the graphics request stream directly in a message from the host processor (via the host bus). Upon receipt by the graphics card, the graphics request stream is written to the graphics card memory.

In yet another embodiment the method the graphics request stream is written through the host processor's write combing buffer.

Brief Description of the Drawings

The foregoing and other objects and advantages of the invention will be appreciated more fully from the following further description thereof with reference to the accompanying drawings wherein:

Fig. 1 is block diagram of a prior art system for placing a graphics request stream into the cache of the host processor.

Fig. 2 is a flow chart of the method used in transferring a graphics request stream onto a graphics accelerator in a prior art system.

Fig. 3 is a block schematic of a graphics card in which a preferred embodiment of the invention may be implemented.

Fig. 4 is a flow chart of a preferred method for transporting a graphics request to direct burst memory of a graphics card.

Fig. 5 is a block diagram of a system in which preferred methods for transferring graphics requests to the graphics card can be implemented.

Fig. 6 is a flow chart of a preferred method of transmitting a graphics request stream to a graphics card.

Detailed Description of the Embodiments

In the following description and claims, the term "graphics request stream" shall refer to multiple instructions which are in a format which is understood by and which may be processed by a graphics card to form a graphical image which can be displayed. In accordance with a preferred embodiment of the invention, a graphics request stream may be transferred directly from a host processor to a memory location on a graphics accelerator card ("graphics card" or "accelerator"). Fig. 3. shows an accelerator **400** which is utilized in a preferred embodiment of the invention. The accelerator **400** is a peripheral component interconnect "PCI" peripheral for a personal computer and connects to a PCI bus **407**. The accelerator **400** includes a decoder shown as a field programmable gate array (FPGA) **401** which provides a PCI bus interface to a graphics card memory **402**, hereinafter referred to as "directburst memory". The directburst memory **402** preferably is synchronous dynamic random access memory (SDRAM) that is memory mapped as write combining memory format into the host processor memory configuration, thus allowing the host processor to send data directly to the direct burst memory as if the memory were on the host processor. The process of memory mapping is performed upon the boot up of the host processor. A driver associated with the graphics card is activated by the operating system and the driver requests a memory address segment which is associated with the host processor. The driver associates the memory address segment of the host processor with a memory buffer **520** which is a segment of contiguous directburst memory **502** on the graphics card **504** as shown in Fig. 4. The graphics card **504** is composed of the directburt memory **502** and the processing engine **530**. The memory buffer of the directburst memory **502** can accept burst write or multiple word transfers across bus **505**. In a preferred embodiment the directburst memory is thirty-two bits wide.

Graphics commands from a graphics application are translated by a graphics API **506** into a graphics request stream **503** and passed to a write combining buffer **510** of the host processor. The driver in conjunction with the host processor **501** reads the graphic request stream **503** from the write combining buffer **510** built up in memory associated with the host processor and writes it to the memory buffer **520** of the directburst memory **502** through the FPGA. The write combining buffer **510** is not part of cache memory, is not snooped and does not provide data coherency. In a preferred embodiment, there are two sets of write combining registers that make up the write combining buffer **510**. The write combining register sets each can hold eight thirty-two bit quantities and each register set is written to the graphics card in turn when the register set is full under normal conditions. As the graphics request stream is bursted from the registers, it is received at the graphics card as a serial sequence of contiguous thirty-two bit quantities. The FPGA decodes and recognizes that burst writes are being received and generates sequential addresses to the memory buffer of the graphics card **504** as it writes each 32-bit quantity to the 32-bit wide memory. It should be understood to one skilled in the art that other decoders implementations may be substituted for the FPGA. Because write combining memory has weak ordering semantics, the ordering may not be maintained for the graphics request stream when it is sent from the write combining registers to the graphics card. However, since each instruction of the graphics request stream has an associated address and the graphics card memory is random access memory (RAM), the ordering is resolved by the FPGA and RAM memory when each address of the graphics request stream is associated with the memory space for that address.

Returning to Fig. 3, the FPGA **401** also connects to a FIFO (First-in First-out) set of registers **404** which connects to a set of digital signal processing chips (DSPs) **403**. The FPGA **401** contains a DMA (Direct Memory Access) engine (not shown) which has a DMA channel **404** that is dedicated to moving data from the directburst memory **402** to the FIFO **408**. In the preferred embodiment, the memory buffer of the directburst memory is double buffered so that one buffer can be under construction by the driver while the contents of the companion buffer are being copied to the FIFO by the DMA engine through the DMA channel. The DSPs then employ internal DMA channels to move the data from the

FIFO into the DSPs. There are six such DSP chips **403** in the preferred embodiment. These six DSP chips make up what is known as the request DSPs. The request DSPs perform the geometry acceleration on the graphics request stream. The geometry stage processing performed by the request DSPs **403** first transforms polygons of three dimensional objects into polygons that can be drawn on a computer screen, then calculates the lighting characteristics, and finally generates a coordinate definition in three dimensions for each polygon. A second DSP chip known as a sequencer DSP **405** strings the processed requests together in the proper order from the request DSPs **403** and passes strings to a rendering engine **406** for eventual display by a display screen (not shown). The rendering stage performed by the rendering engine converts polygon information to pixels for display. It involves applying shading, texture maps, and atmospheric/special effects to the polygon information provided by the geometry stage. Additional explanation of the graphics card is provided in United States Provisional Patent Application entitled WIDE INSTRUCTION WORD GRAPHICS PROCESSOR, Serial No. 60/093,165, filed July 17, 1998 and bearing attorney docket number 1247/134.

Fig. 5 is flow chart of the steps taken in configuring the host processor to transfer graphics request streams to the graphics card. Host processors, such as the PentiumPro™ microprocessor having a P6 bus (available from Intel Corporation of Santa Clara, California) are provided with the ability to assign a memory address to a memory location which is outside of RAM memory associated with the host processor. The method first assigns an address of the host processor to memory from the graphics card. **(Step 602)** The driver associated with the graphics card asks the operating system to provide a block of memory addresses that are equivalent to the memory size of the directburst memory on the graphics card. In one embodiment, the host processor has a limited number of memory address locations and the host processor has designated memory addresses allocated for external devices which have associated memory.

When a graphics request stream is sent to the host processor, the host processor recognizes that the graphics request stream should be sent to the memory located on the graphics card based upon the address for the graphics request stream.(**Step 604**) The host processor fills a write combining buffer with the graphics request stream until the write combining buffer is full. The host processor then sends the graphic request streams directly to the direct burst memory of the graphics card (**Step 606**).

Fig. 6 is a flow chart of a preferred method of transmitting a graphics request stream to a graphics card. In response to an application level program that requests a graphics display, the preferred method eliminates the need to transfer the request to the cached main memory of the host processor by transmitting the requests from the CPU in an efficient manner. Specifically, in step **702**, the application level program makes a call through the host processor via API calls for graphics rendering. In one embodiment, the API **506** is the OpenGL™ API. OpenGL is an industry standard 3D graphics processing library that allows computer programmers to draw sophisticated graphics on the computer video screen by making calls to OpenGL graphics library commands. The API commands are then translated by a driver program which formats the API commands into an graphics request stream that is understood by the graphics card. Once the API calls **506** are translated, the graphics request stream, **503** is directed to the graphics card **504** (step **704**).

The graphic request stream is written directly by the processor in step **706**, to the directburst memory **502** on the graphics card. The host processor **501** has the directburst memory **502** mapped into the host processor. Additionally, for increased speed, the direct burst memory **502** on the video graphics card **504** can accept burst write transfers which traverse the processor bus and the PCI bus **505** only once (step **708**). This consequently frees up the cached main memory for other memory intensive calculations and reduces the total amount of reads and writes for the host processor. Write combining buffers in the host processor, as well as in the PCI bus interface device (not shown), ensure that the writes transpire across the PCI bus as large efficient bursts. Once the graphics request stream is stored in the graphics card's memory, the graphics request stream may be placed in a FIFO for access by the DSPs. The graphic request streams are processed in the request DSPs and in the rendering engine of the chip in step **710**. In step **712**, the output is then sent to a display device for display.

Although various exemplary embodiments of the invention have been disclosed, it should be apparent to those skilled in the art that various changes and modifications can be made which will achieve some of the advantages of the invention without departing from the true scope of the invention. These and other obvious modifications are intended to be covered by the appended claims.

We claim:

1. A method of transferring a graphics request stream to a graphics card via a host bus, the graphics card having graphics card memory, the method comprising:

receiving the graphics request stream from the host processor via the host bus, the graphics request stream traversing the host bus no more than once; and

writing the graphics request stream to the graphics card memory.

2. The method according to claim 1, further comprising the step of :

recognizing each address within the graphics request stream;

wherein the graphics request stream is written to the corresponding address in the graphics card memory.

3. The method according to claim 2, wherein the graphics request stream is in order after the step of writing.

4. The method according to claim 1, wherein in the step of receiving, the graphics request stream is initially located in a write combining register.

5. The method according to claim 1, wherein the graphics card memory is random access memory.

6. The method according to claim 1, wherein the random access memory is synchronous dynamic access memory.

7. A method of transferring a graphics request stream from a host processor to a graphics card, the method comprising:

writing the graphics request stream to the host processor;

reading the graphics request stream from the host processor;

traversing a system bus with the graphics request stream no more than once;

writing the graphics request stream to a memory location on the graphics card.

8. The method according to claim 7, wherein in the step of writing, the graphics request stream is written to a write combining register in the host processor.

9. The method according to claim 7, wherein the memory location on the graphics card is random access memory

10. The method according to claim 9, wherein a field programmable gate array directs each instruction of the graphics request stream to an associated address in the random access memory.

11. The method according to claim 1, wherein the host processor has a system for assigning addresses to memory, the method further comprising the step of:
assigning an address to the memory of the graphics card.

12. The method according to claim 11, wherein in the step of assigning the address to memory the memory is assigned as write combining memory.

13. The method according to claim 7, wherein each instruction of the graphics request stream is associated with an address on the graphics card and in the step of writing, the graphics request stream is written in bursts, in which, multiple instructions of the graphics request stream are written to the graphics card at the same time.

14. A method of transferring a graphics request stream from a host processor to a graphics card, the method comprising:
creating a memory buffer on the graphics card;
writing the graphics request stream to the host processor;

forwarding the graphics request stream from the host processor to the graphics card via a bus;

the graphics request stream, traversing the bus no more than once; and

writing the graphics request stream to the memory buffer.

15. The method according to claim 14, wherein creating a memory buffer is executed by the host processor.

16. The method according to claim 14 further comprising the step of :

a host processor receiving a command to draw a graphic from an application program;

accessing a graphics application program interface for calling graphics functions in response to the command to form a graphics request stream.

17. The method according to claim 15 wherein the graphics application program interface is OpenGL.

18. A method of transferring a graphics request stream to a graphics card, the method comprising:

accessing driver software to allow a graphics card to interpret graphics request streams in response to application level programs that request an image to be drawn;

defining a graphics request stream in a buffer associated with a host processor;

traversing the bus no more than once with the graphics request stream; and

writing graphic request stream to the memory on the graphics card in contiguous order.

19. A method according to claim 1 wherein the host processor is a PentiumPro processor.

20. The method according to claim 1, wherein the host bus is a P6 bus.

21. The method as defined by claim 18 wherein the host processor has cache memory and the graphics request stream is not stored in the cache memory at any time before the graphics request stream is written to the graphics card memory.

22. The method according to claim 18 further comprising the step of:

accessing an address of the graphics request stream to determine a destination for the graphics request stream.

23. A method for transferring a graphics request stream from a host processor to a graphics card, the method comprising:

the host processor assigning a memory address to graphics memory on the graphics card;

identifying data received by the host processor as a graphics request stream; and

sending the graphics request stream from the host processor to the memory address of the graphics memory on the graphics card.

24. A system for reordering a graphics request stream that is bursted in a write combining memory format to a graphics card, the system comprising:

addressable memory for receiving instructions of the graphics request stream; and

a decoder for recognizing an address associated with the instructions of the graphics request stream and forwarding the instructions to the addressable memory.

25. The system according to claim 24, wherein the decoder is a field gate programmable array.

26. The system according to claim 24, wherein the addressable memory is configured as a buffer.

27. The system according to claim 24, wherein the instructions are ordered by the decoder so that the instructions are placed in contiguous addressable memory locations.

28. The system according to claim 24, wherein the addressable memory is random access memory.

29. The system according to claim 28, wherein the random access memory is synchronous dynamic random access memory.

92201

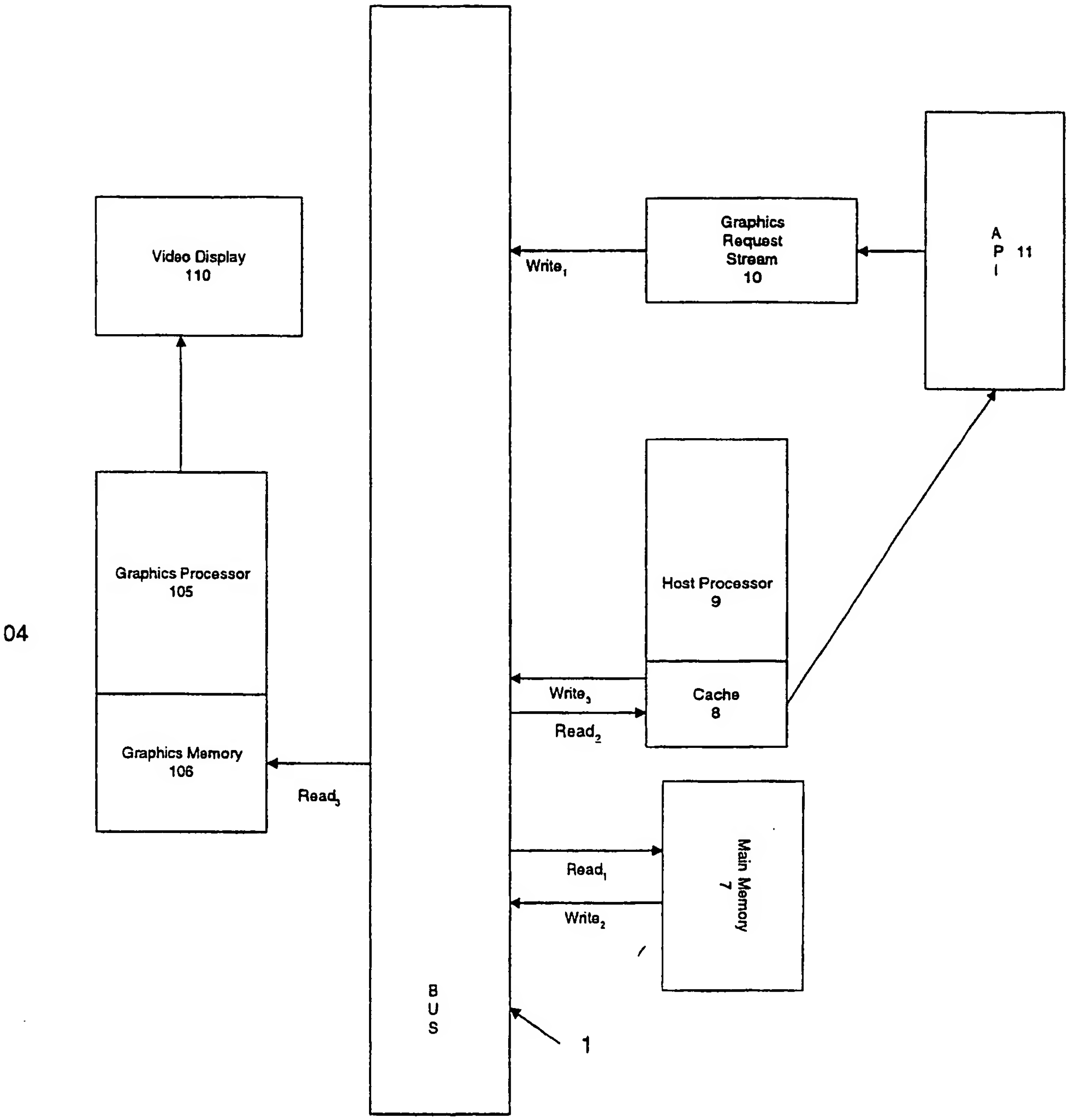


Fig. 1
Prior Art

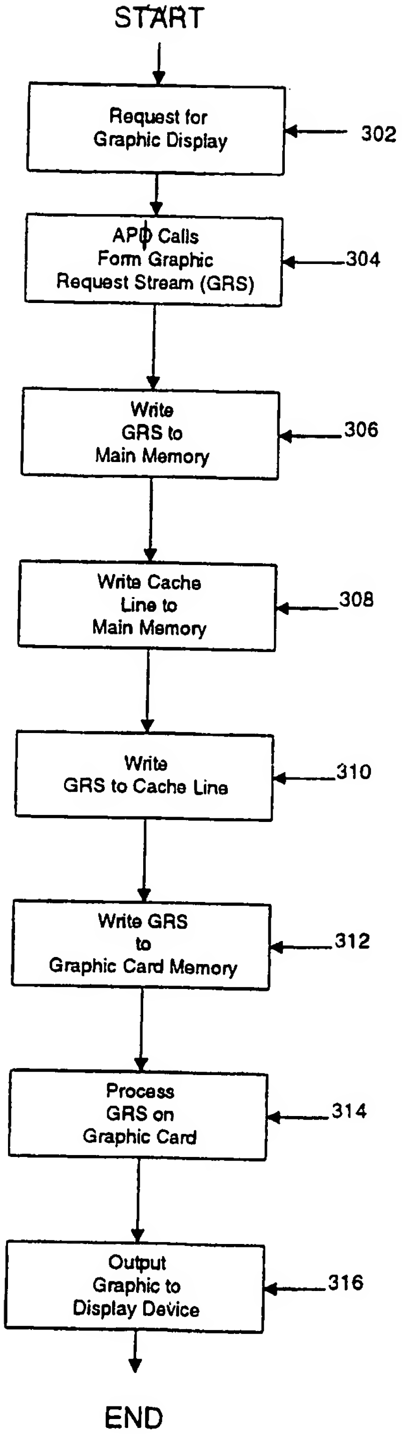

START
Request for Graphic Display
302
APD Calls Form Graphic Request Stream (GRS)
304
Write GRS to Main Memory
306
Write Cache Line to Main Memory
308
Write GRS to Cache Line
310
Write GRS to Graphic Card Memory
312
Process GRS on Graphic Card
314
Output Graphic to Display Device
316
END

Fig. 2

Prior Art

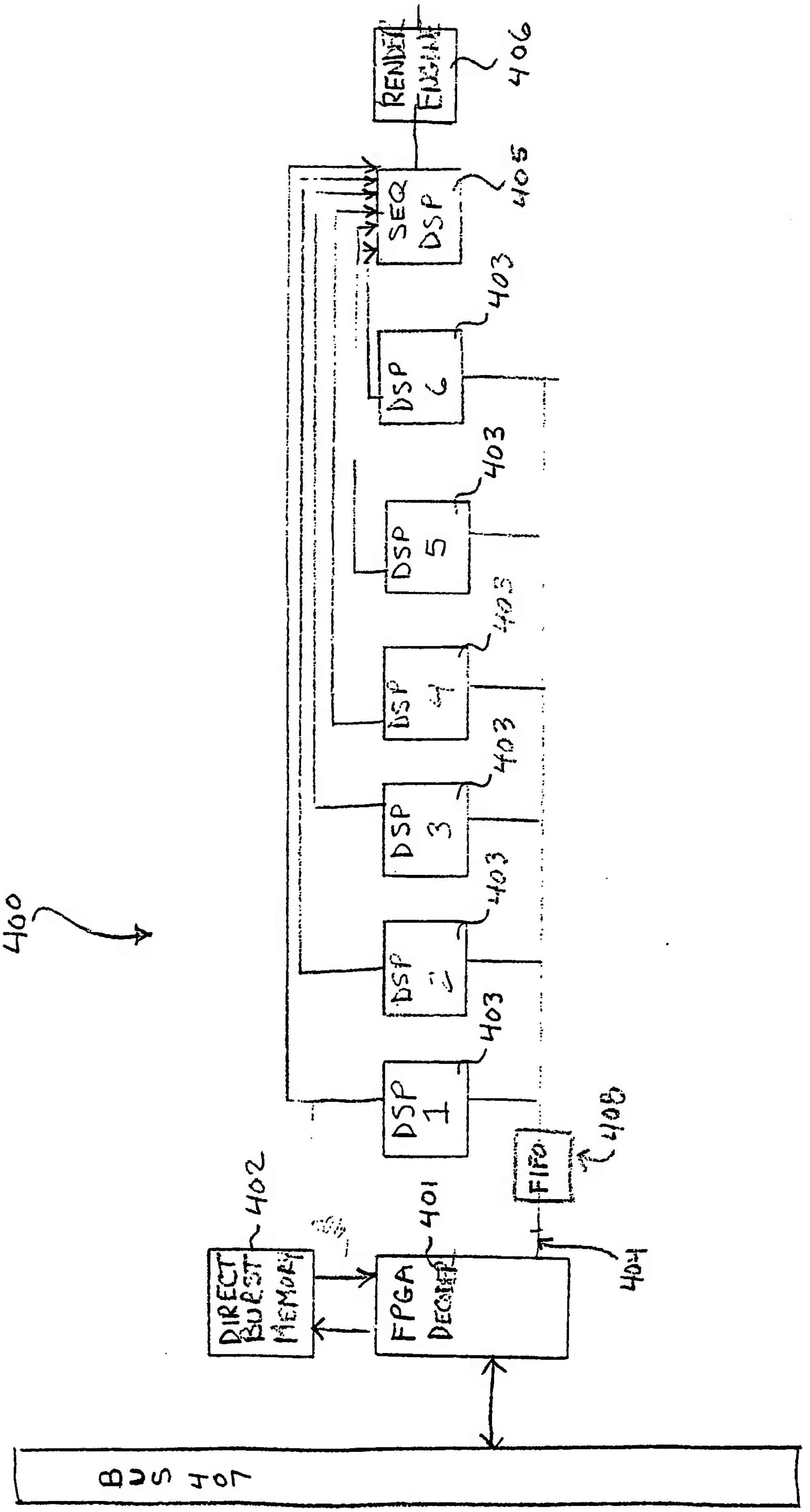

400
DIRECT BURST MEMORY
402
FPGA DECODER
401
FIFO
408
404
DSP 1
DSP 2
DSP 3
DSP 4
DSP 5
DSP 6
403
SEQ DSP
405
RENDER ENGINE
406
BUS 407

FIG. 3

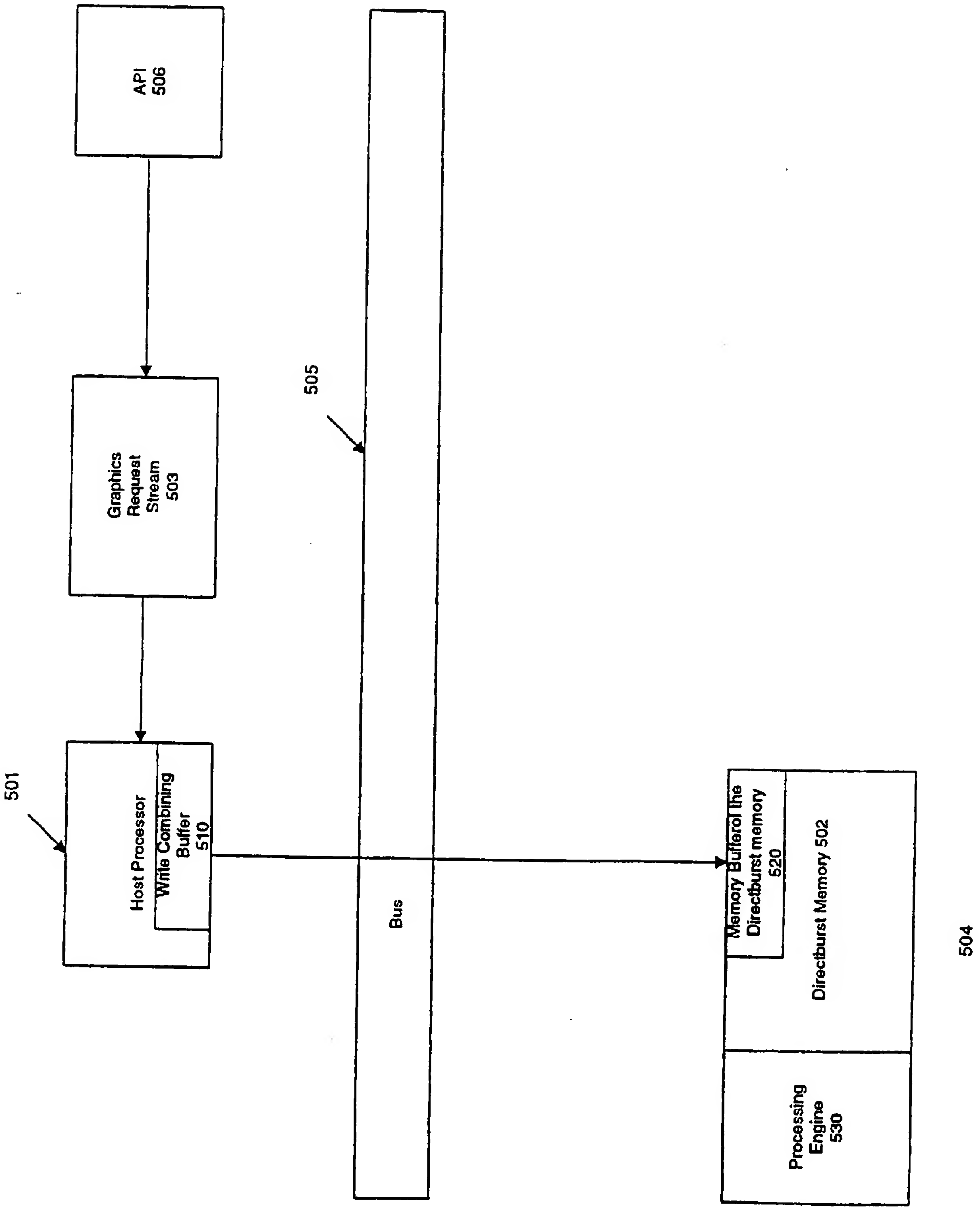

API
506
Graphics
Request
Stream
503
501
Host Processor
Write Combining
Buffer
510
505
Bus
Memory Bufferof the
Directburst memory
520
Directburst Memory 502
Processing
Engine
530
504

Fig. 4

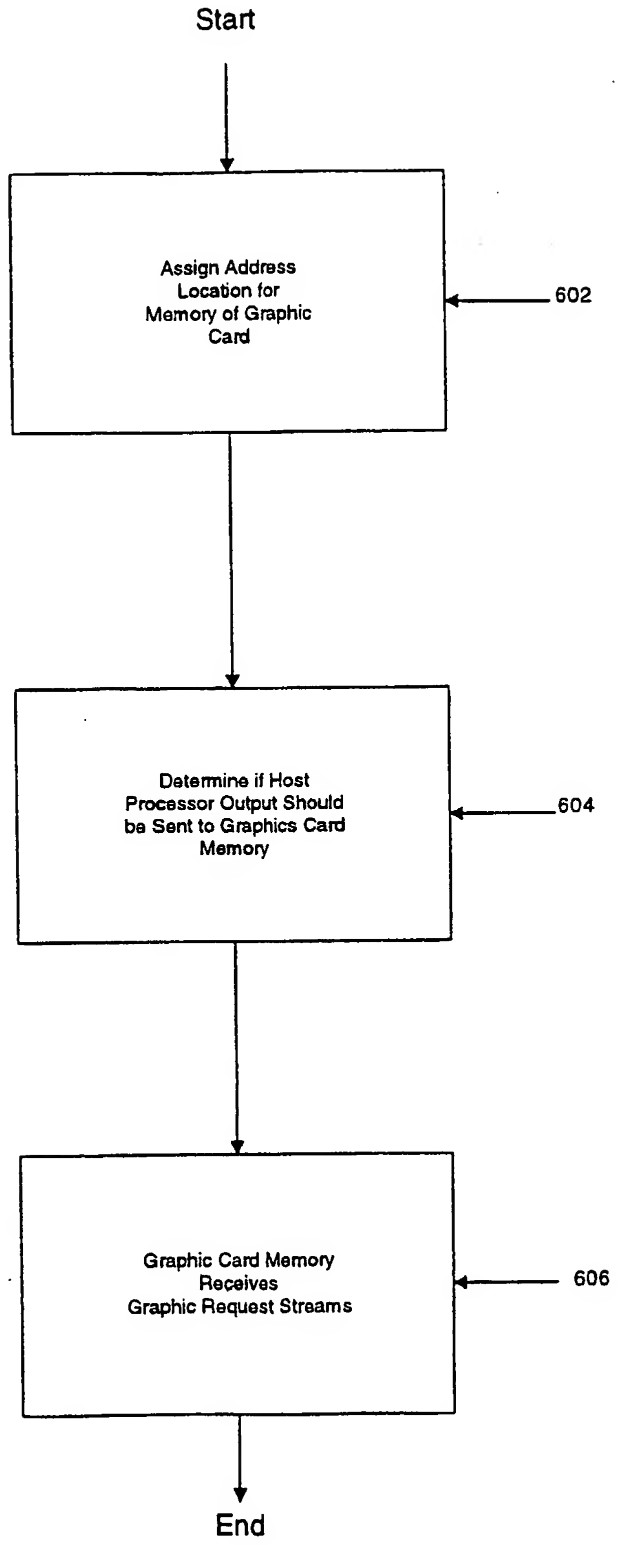

Start
Assign Address Location for Memory of Graphic Card
602
Determine if Host Processor Output Should be Sent to Graphics Card Memory
604
Graphic Card Memory Receives Graphic Request Streams
606
End

Fig. 5

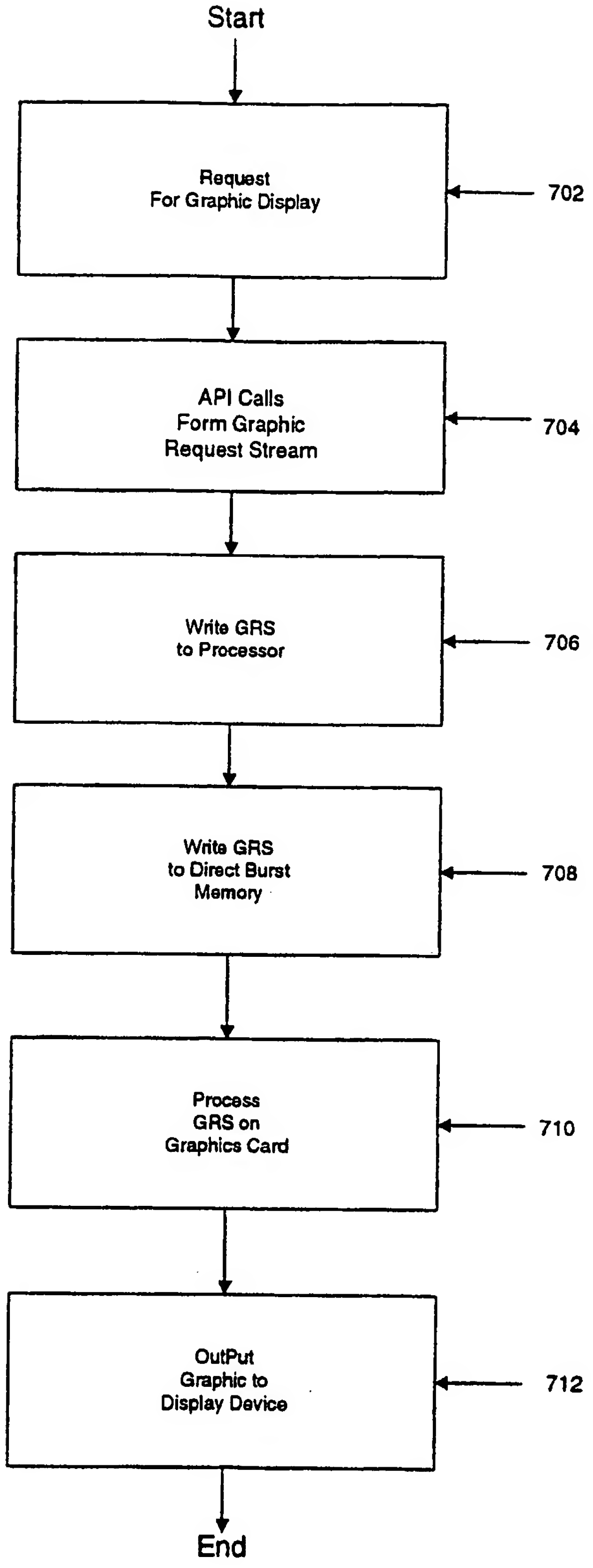

Start
Request
For Graphic Display
702
API Calls
Form Graphic
Request Stream
704
Write GRS
to Processor
706
Write GRS
to Direct Burst
Memory
708
Process
GRS on
Graphics Card
710
OutPut
Graphic to
Display Device
712
End

Fig. 6

INTERNATIONAL SEARCH REPORT

Interr nal Application No
PCT/US 99/14889

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/38 G06F13/12 G06T1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 717 348 A (MOTOROLA INC) 19 June 1996 (1996-06-19)	1-3,5,6, 23-29
Y	column 1, line 28 -column 2, line 26	4,7-14, 16,18-22
A	column 2, line 52 -column 3, line 3 column 6, line 3 - line 56 column 12, line 11 -column 13, line 56 ---	4,8,12
Y	WO 93 04462 A (DIGITAL EQUIPMENT CORP) 4 March 1993 (1993-03-04) page 5, line 4 - line 28 page 7, line 9 - line 21 page 8, line 14 -page 9, line 11 page 13, line 22 -page 14, line 14 page 26, line 11 -page 29, line 37 --- -/--	4,7-14, 16,18-22

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 29 October 1999	Date of mailing of the international search report 05/11/1999
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Daskalakis, T

International Application No
PCT/US 99/14889

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 630 075 A (GLEW ANDREW F ET AL) 13 May 1997 (1997-05-13) the whole document ---	4,8,12
A	US 5 619 658 A (ROSENTHAL DAVID S H ET AL) 8 April 1997 (1997-04-08) column 4, line 27 -column 5, line 34 column 6, line 34 -column 9, line 23 ---	15-17
A	EP 0 483 970 A (IBM) 6 May 1992 (1992-05-06) column 2, line 55 -column 3, line 35 column 5, line 44 - line 55 column 7, line 53 -column 15, line 3 -----	1,7,14, 18,23,24

1

INTERNATIONAL SEARCH REPORT

.ormation on patent family members

Intern nal Application No
PCT/US 99/14889

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0717348 A	19-06-1996	JP 8339290 A	24-12-1996
		US 5889948 A	30-03-1999
WO 9304462 A	04-03-1993	US 5313577 A	17-05-1994
		US 5321806 A	14-06-1994
		AU 2553592 A	16-03-1993
US 5630075 A	13-05-1997	NONE	
US 5619658 A	08-04-1997	NONE	
EP 0483970 A	06-05-1992	WO 9208192 A	14-05-1992
		HU 65534 A,B	28-06-1994
		IL 99744 A	26-05-1995
		JP 4289955 A	14-10-1992
		PL 166975 B	31-07-1995
		SK 41393 A	07-07-1993
		US 5539896 A	23-07-1996

CORRECTED VERSION*

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 :
G06F 9/38, 13/12, G06T 1/00

A1

(11) International Publication Number: WO 00/00887

(43) International Publication Date: 6 January 2000 (06.01.00)

(21) International Application Number: PCT/US99/14889

(22) International Filing Date: 30 June 1999 (30.06.99)

(30) Priority Data:
60/091,401 30 June 1998 (30.06.98) US

(71) Applicant: INTERGRAPH CORPORATION [US/US]; One Madison Industrial Park, Huntsville, AL 35894–0001 (US).

(72) Inventors: TERRY, Joseph, Clay; 3219 Highland Plaza, Huntsville, AL 35801 (US). KIRKLAND, Dale, L.; 106 Spring Water Drive, Madison, AL 35758 (US). CONKLIN, Steve; 514 Mill Road, Madison, AL 35758 (US). QUINN, Matthew, C.; 201 Water Hill Road, Madison, AL 35758 (US).

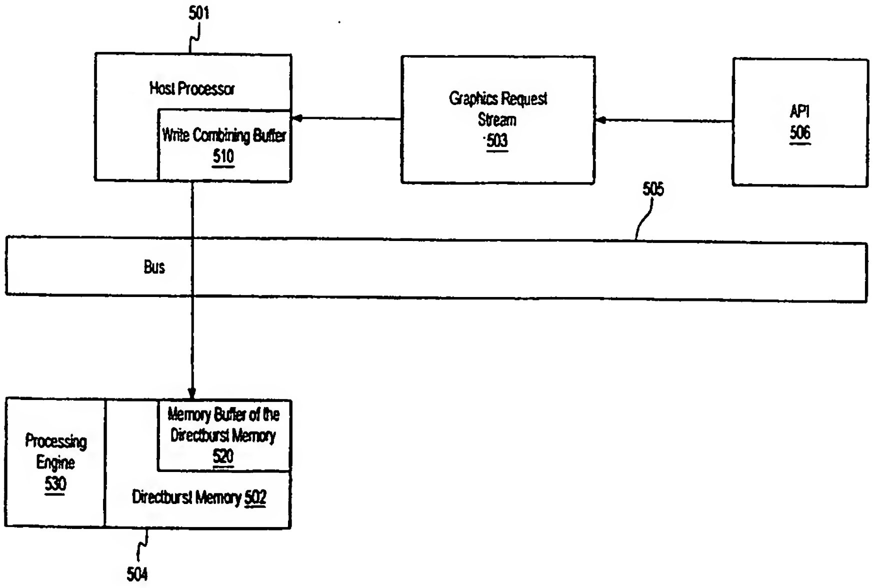
(74) Agents: SUNSTEIN, Bruce, D. et al.; Bromberg & Sunstein LLP, 125 Summer Street, Boston, MA 02110–1618 (US).

(81) Designated States: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published

With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD AND APPARATUS FOR TRANSPORTING INFORMATION TO A GRAPHIC ACCELERATOR CARD


501
Host Processor
Write Combining Buffer 510
Graphics Request Stream 503
API 506
505
Bus
Processing Engine 530
Memory Buffer of the Directburst Memory 520
Directburst Memory 502
504

(57) Abstract

A graphics request stream is transferred from a host processor to a graphics card via a host bus so that the stream traverses the host bus no more than once. To that end, the graphics card has a graphics card memory, and the host processor has a host memory configured in a first memory configuration. The graphics card memory may be configured in the first memory configuration, and the graphics request stream is received directly in a message from the host processor (via the host bus). Upon receipt by the graphics card, the graphics request stream is written to the graphics card memory.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

Method and Apparatus for Transporting Information to a Graphic Accelerator Card

Field of the Invention

The present invention is related to graphics accelerator cards and, more particularly, involves the use of memory on graphics accelerator cards.

Background of the Invention

Typical computer systems employ a graphics accelerator card for enhancing the resolution and the display of graphics. The display of graphics requires a two part process, rendering and geometry acceleration. In prior art graphics cards, the geometry phase was performed by the central processing unit (CPU) of the computer system while the rendering phase was performed by the graphics card. The (CPU) is often referred to as a host processor. This often overloaded the CPU, since graphics were vying for processor time with external applications. Currently, high-end graphics cards have been configured to perform both the rendering phase and the geometry phase. This system improves performance and graphic rendering because the central processing unit is free to perform other processes while the graphics are being processed on the graphics card.

Although performance is increased during processing by having the graphics card perform both rendering and geometry acceleration, the graphics request must still be sent to the graphics card through the CPU which involves significant memory swaps between RAM memory and cache memory associated with the CPU.

See Fig.1 for a schematic diagram of the components involved in an exemplary prior art graphics card. Fig. 1 shows a host processor 9 of a computer system which is connected to a bus 1. The bus 1 is used for transporting information to and from various components of the computer system, including main memory 7. The host processor 9 receives a request from an application level program to create a graphics display. The request may be in the form of a group of instructions which accesses an application program interface ("API") 11. The

API converts the instructions into a graphics request stream **10** which is capable of being understood by the graphics accelerator. The graphics request stream **10** is transmitted to a cache **8** associated with the host processor, and placed into a cache line via bus **1**. The graphics request stream is transported from the cache **8** across the bus **1** and deposited in a graphics memory location **106** of the graphics card **104**. The graphics request stream **10** is processed by a graphics processor **105** and then sent to a display device.

Figure 2 shows a prior art method of receiving the graphics request and transporting the graphics request stream to the graphics accelerator card for processing. The process begins at step **302**, in which an application level program makes a request for a graphics display. This causes the appropriate functions of the API **11** to be called. The result of the API functions form a graphics request stream **10** based on the request from the application level program in step **304**.

The host processor **9** writes the graphics request stream **10** to main memory **7** in step **306**, which requires the graphics request stream to pass across the system bus. Cache read and write is indicated by a subscript numeral in Fig. 1. Because the position in main memory **7** that is written to is typically not in the cache **8**, and the cache line usually has data in it that is not coherent with main memory **7**, a cache line swap must take place. This involves writing the current cache line contents into an associated main memory location **7**, (step **308**), and writing the newly addressed cache line **12** having the graphics request stream into the cache (step **310**). Thus, writing the graphics request stream to the cache of the CPU requires the graphics request stream to pass across the system bus twice. Once the data of the graphics request stream **10** is cached in the cache memory, it still must be moved into the graphics system before rendering can occur, thus requiring a third crossing of the system bus, (step **312**). To do this, a graphics processor **105** on the graphics card **104** is controlled by driver software. The driver software causes the host processor to read the graphics request stream **10** from the cached memory **8**, and then passes the graphics request stream to the graphics processor **105** of the graphics card which writes it into a memory location **106** for processing (step **314**). Once initiated, the graphics processor **105** proceeds without further intervention by the CPU **9**, and the processed graphics request stream is displayed by a display device, (step **316**).

In summary, each word of data of the graphics request stream that is moved into the graphics accelerator requires two transactions for storage in cache memory, and one transaction to move it from cache memory 8 to the graphics pipeline **106**. Processing data in this way thus requires at least three read/writes across the system bus, consequently reducing the rendering speed to no faster than about thirty-three percent of the system bus rate.

Summary of the Invention

In accordance with one aspect of the invention, a graphics request stream is transferred from a host processor to a graphics card via a host bus so that the stream traverses the bus no more than once. To that end, the graphics card has a graphics card memory, and the host processor has an address system for addressing the graphics card memory. In accordance with preferred embodiments of the invention, the graphics card receives the graphics request stream directly in a message from the host processor (via the host bus). Upon receipt by the graphics card, the graphics request stream is written to the graphics card memory.

In yet another embodiment the method the graphics request stream is written through the host processor's write combing buffer.

Brief Description of the Drawings

The foregoing and other objects and advantages of the invention will be appreciated more fully from the following further description thereof with reference to the accompanying drawings wherein:

Fig. 1 is block diagram of a prior art system for placing a graphics request stream into the cache of the host processor.

Fig. 2 is a flow chart of the method used in transferring a graphics request stream onto a graphics accelerator in a prior art system.

Fig. 3 is a block schematic of a graphics card in which a preferred embodiment of the invention may be implemented.

Fig. 4 is a flow chart of a preferred method for transporting a graphics request to direct burst memory of a graphics card.

Fig. 5 is a block diagram of a system in which preferred methods for transferring graphics requests to the graphics card can be implemented.

Fig. 6 is a flow chart of a preferred method of transmitting a graphics request stream to a graphics card.

Detailed Description of the Embodiments

In the following description and claims, the term "graphics request stream" shall refer to multiple instructions which are in a format which is understood by and which may be processed by a graphics card to form a graphical image which can be displayed. In accordance with a preferred embodiment of the invention, a graphics request stream may be transferred directly from a host processor to a memory location on a graphics accelerator card ("graphics card" or "accelerator"). Fig. 3. shows an accelerator **400** which is utilized in a preferred embodiment of the invention. The accelerator **400** is a peripheral component interconnect "PCI" peripheral for a personal computer and connects to a PCI bus **407**. The accelerator **400** includes a decoder shown as a field programmable gate array (FPGA) **401** which provides a PCI bus interface to a graphics card memory **402**, hereinafter referred to as "directburst memory". The directburst memory **402** preferably is synchronous dynamic random access memory (SDRAM) that is memory mapped as write combining memory format into the host processor memory configuration, thus allowing the host processor to send data directly to the direct burst memory as if the memory were on the host processor. The process of memory mapping is performed upon the boot up of the host processor. A driver associated with the graphics card is activated by the operating system and the driver requests a memory address segment which is associated with the host processor. The driver associates the memory address segment of the host processor with a memory buffer **520** which is a segment of contiguous directburst memory **502** on the graphics card **504** as shown in Fig. 4. The graphics card **504** is composed of the directburt memory **502** and the processing engine **530**. The memory buffer of the directburst memory **502** can accept burst write or multiple word transfers across bus **505**. In a preferred embodiment the directburst memory is thirty-two bits wide.

Graphics commands from a graphics application are translated by a graphics API **506** into a graphics request stream **503** and passed to a write combining buffer **510** of the host processor. The driver in conjunction with the host processor **501** reads the graphic request stream **503** from the write combining buffer **510** built up in memory associated with the host processor and writes it to the memory buffer **520** of the directburst memory **502** through the FPGA. The write combining buffer **510** is not part of cache memory, is not snooped and does not provide data coherency. In a preferred embodiment, there are two sets of write combining registers that make up the write combining buffer **510**. The write combining register sets each can hold eight thirty-two bit quantities and each register set is written to the graphics card in turn when the register set is full under normal conditions. As the graphics request stream is bursted from the registers, it is received at the graphics card as a serial sequence of contiguous thirty-two bit quantities. The FPGA decodes and recognizes that burst writes are being received and generates sequential addresses to the memory buffer of the graphics card **504** as it writes each 32-bit quantity to the 32-bit wide memory. It should be understood to one skilled in the art that other decoders implementations may be substituted for the FPGA. Because write combining memory has weak ordering semantics, the ordering may not be maintained for the graphics request stream when it is sent from the write combining registers to the graphics card. However, since each instruction of the graphics request stream has an associated address and the graphics card memory is random access memory (RAM), the ordering is resolved by the FPGA and RAM memory when each address of the graphics request stream is associated with the memory space for that address.

Returning to Fig. 3, the FPGA **401** also connects to a FIFO (First-in First-out) set of registers **404** which connects to a set of digital signal processing chips (DSPs) **403**. The FPGA **401** contains a DMA (Direct Memory Access) engine (not shown) which has a DMA channel **404** that is dedicated to moving data from the directburst memory **402** to the FIFO **408**. In the preferred embodiment, the memory buffer of the directburst memory is double buffered so that one buffer can be under construction by the driver while the contents of the companion buffer are being copied to the FIFO by the DMA engine through the DMA channel. The DSPs then employ internal DMA channels to move the data from the

FIFO into the DSPs. There are six such DSP chips **403** in the preferred embodiment. These six DSP chips make up what is known as the request DSPs. The request DSPs perform the geometry acceleration on the graphics request stream. The geometry stage processing performed by the request DSPs **403** first transforms polygons of three dimensional objects into polygons that can be drawn on a computer screen, then calculates the lighting characteristics, and finally generates a coordinate definition in three dimensions for each polygon. A second DSP chip known as a sequencer DSP **405** strings the processed requests together in the proper order from the request DSPs **403** and passes strings to a rendering engine **406** for eventual display by a display screen (not shown). The rendering stage performed by the rendering engine converts polygon information to pixels for display. It involves applying shading, texture maps, and atmospheric/special effects to the polygon information provided by the geometry stage. Additional explanation of the graphics card is provided in United States Provisional Patent Application entitled WIDE INSTRUCTION WORD GRAPHICS PROCESSOR, Serial No. 60/093,165, filed July 17, 1998 and bearing attorney docket number 1247/134.

Fig. 5 is flow chart of the steps taken in configuring the host processor to transfer graphics request streams to the graphics card. Host processors, such as the PentiumPro™ microprocessor having a P6 bus (available from Intel Corporation of Santa Clara, California) are provided with the ability to assign a memory address to a memory location which is outside of RAM memory associated with the host processor. The method first assigns an address of the host processor to memory from the graphics card. **(Step 602)** The driver associated with the graphics card asks the operating system to provide a block of memory addresses that are equivalent to the memory size of the directburst memory on the graphics card. In one embodiment, the host processor has a limited number of memory address locations and the host processor has designated memory addresses allocated for external devices which have associated memory.

When a graphics request stream is sent to the host processor, the host processor recognizes that the graphics request stream should be sent to the memory located on the graphics card based upon the address for the graphics request stream.**(Step 604)** The host processor fills a write combining buffer with the graphics request stream until the write combining buffer is full. The host processor then sends the graphic request streams directly to the direct burst memory of the graphics card **(Step 606)**.

Fig. 6 is a flow chart of a preferred method of transmitting a graphics request stream to a graphics card. In response to an application level program that requests a graphics display, the preferred method eliminates the need to transfer the request to the cached main memory of the host processor by transmitting the requests from the CPU in an efficient manner. Specifically, in step **702**, the application level program makes a call through the host processor via API calls for graphics rendering. In one embodiment, the API **506** is the OpenGL™ API. OpenGL is an industry standard 3D graphics processing library that allows computer programmers to draw sophisticated graphics on the computer video screen by making calls to OpenGL graphics library commands. The API commands are then translated by a driver program which formats the API commands into an graphics request stream that is understood by the graphics card. Once the API calls **506** are translated, the graphics request stream, **503** is directed to the graphics card **504** (step **704**).

The graphic request stream is written directly by the processor in step **706**, to the directburst memory **502** on the graphics card. The host processor **501** has the directburst memory **502** mapped into the host processor. Additionally, for increased speed, the direct burst memory **502** on the video graphics card **504** can accept burst write transfers which traverse the processor bus and the PCI bus **505** only once (step **708**). This consequently frees up the cached main memory for other memory intensive calculations and reduces the total amount of reads and writes for the host processor. Write combining buffers in the host processor, as well as in the PCI bus interface device (not shown), ensure that the writes transpire across the PCI bus as large efficient bursts. Once the graphics request stream is stored in the graphics card's memory, the graphics request stream may be placed in a FIFO for access by the DSPs. The graphic request streams are processed in the request DSPs and in the rendering engine of the chip in step **710**. In step **712**, the output is then sent to a display device for display.

Although various exemplary embodiments of the invention have been disclosed, it should be apparent to those skilled in the art that various changes and modifications can be made which will achieve some of the advantages of the invention without departing from the true scope of the invention. These and other obvious modifications are intended to be covered by the appended claims.

We claim:

1. A method of transferring a graphics request stream to a graphics card via a host bus, the graphics card having graphics card memory, the method comprising:

receiving the graphics request stream from the host processor via the host bus, the graphics request stream traversing the host bus no more than once; and

writing the graphics request stream to the graphics card memory.

2. The method according to claim 1, further comprising the step of :

recognizing each address within the graphics request stream;

wherein the graphics request stream is written to the corresponding address in the graphics card memory.

3. The method according to claim 2, wherein the graphics request stream is in order after the step of writing.

4. The method according to claim 1, wherein in the step of receiving, the graphics request stream is initially located in a write combining register.

5. The method according to claim 1, wherein the graphics card memory is random access memory.

6. The method according to claim 1, wherein the random access memory is synchronous dynamic access memory.

7. A method of transferring a graphics request stream from a host processor to a graphics card, the method comprising:

writing the graphics request stream to the host processor;

reading the graphics request stream from the host processor;

traversing a system bus with the graphics request stream no more than once;

writing the graphics request stream to a memory location on the graphics card.

8. The method according to claim 7, wherein in the step of writing, the graphics request stream is written to a write combining register in the host processor.

9. The method according to claim 7, wherein the memory location on the graphics card is random access memory

10. The method according to claim 9, wherein a field programmable gate array directs each instruction of the graphics request stream to an associated address in the random access memory.

11. The method according to claim 1, wherein the host processor has a system for assigning addresses to memory, the method further comprising the step of:
assigning an address to the memory of the graphics card.

12. The method according to claim 11, wherein in the step of assigning the address to memory the memory is assigned as write combining memory.

13. The method according to claim 7, wherein each instruction of the graphics request stream is associated with an address on the graphics card and in the step of writing, the graphics request stream is written in bursts, in which, multiple instructions of the graphics request stream are written to the graphics card at the same time.

14. A method of transferring a graphics request stream from a host processor to a graphics card, the method comprising:

creating a memory buffer on the graphics card;

writing the graphics request stream to the host processor;

forwarding the graphics request stream from the host processor to the graphics card via a bus;

the graphics request stream, traversing the bus no more than once; and

writing the graphics request stream to the memory buffer.

15. The method according to claim 14, wherein creating a memory buffer is executed by the host processor.

16. The method according to claim 14 further comprising the step of :

a host processor receiving a command to draw a graphic from an application program;

accessing a graphics application program interface for calling graphics functions in response to the command to form a graphics request stream.

17. The method according to claim 15 wherein the graphics application program interface is OpenGL.

18. A method of transferring a graphics request stream to a graphics card, the method comprising:

accessing driver software to allow a graphics card to interpret graphics request streams in response to application level programs that request an image to be drawn;

defining a graphics request stream in a buffer associated with a host processor;

traversing the bus no more than once with the graphics request stream; and

writing graphic request stream to the memory on the graphics card in contiguous order.

19. A method according to claim 1 wherein the host processor is a PentiumPro processor.

20. The method according to claim 1, wherein the host bus is a P6 bus.

21. The method as defined by claim 18 wherein the host processor has cache memory and the graphics request stream is not stored in the cache memory at any time before the graphics request stream is written to the graphics card memory.

22. The method according to claim 18 further comprising the step of:

accessing an address of the graphics request stream to determine a destination for the graphics request stream.

23. A method for transferring a graphics request stream from a host processor to a graphics card, the method comprising:

the host processor assigning a memory address to graphics memory on the graphics card;

identifying data received by the host processor as a graphics request stream; and

sending the graphics request stream from the host processor to the memory address of the graphics memory on the graphics card.

24. A system for reordering a graphics request stream that is bursted in a write combining memory format to a graphics card, the system comprising:

addressable memory for receiving instructions of the graphics request stream; and

a decoder for recognizing an address associated with the instructions of the graphics request stream and forwarding the instructions to the addressable memory.

25. The system according to claim 24, wherein the decoder is a field gate programmable array.

26. The system according to claim 24, wherein the addressable memory is configured as a buffer.

27. The system according to claim 24, wherein the instructions are ordered by the decoder so that the instructions are placed in contiguous addressable memory locations.

28. The system according to claim 24, wherein the addressable memory is random access memory.

29. The system according to claim 28, wherein the random access memory is synchronous dynamic random access memory.

92201

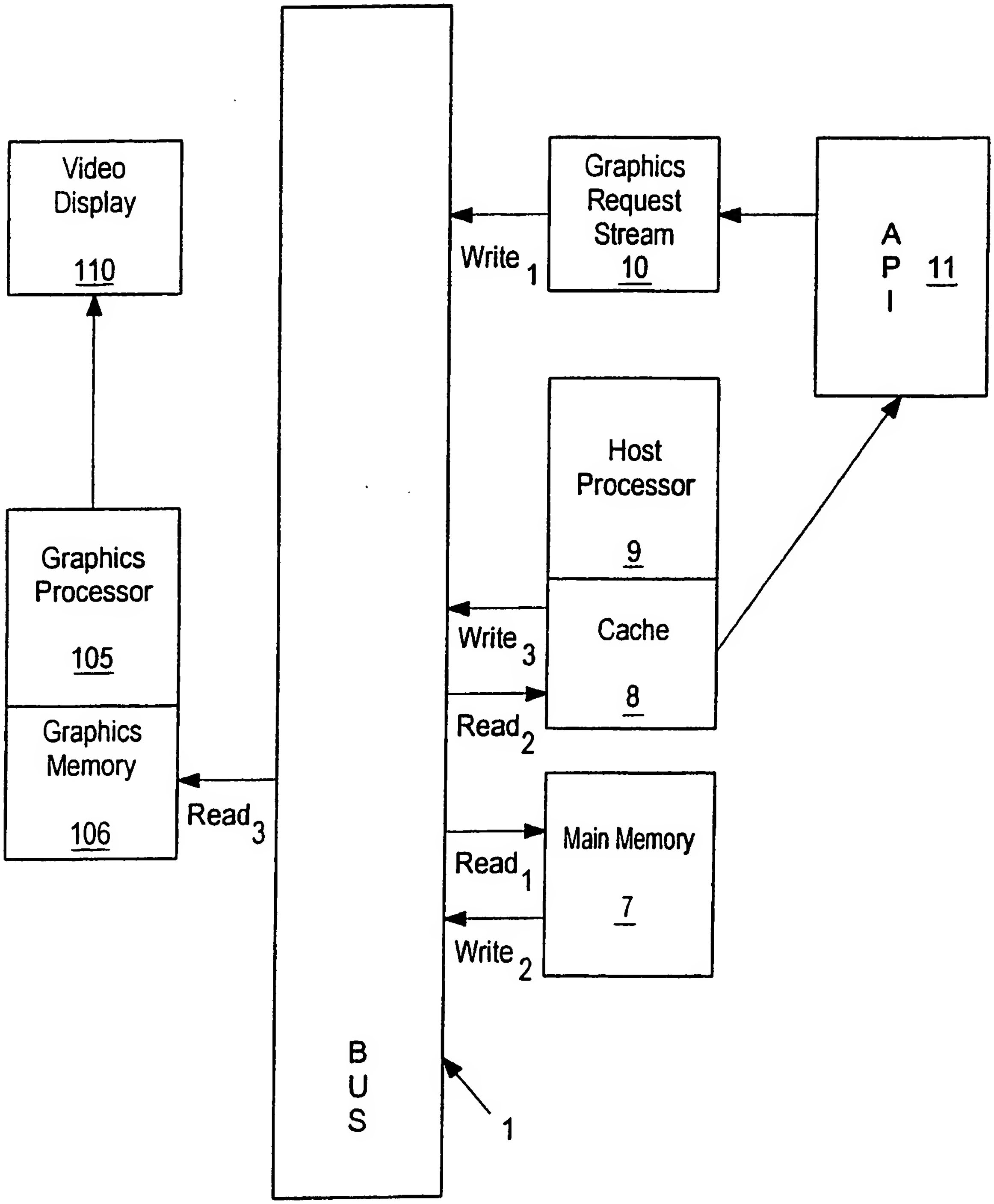


FIG. 1
Prior Art

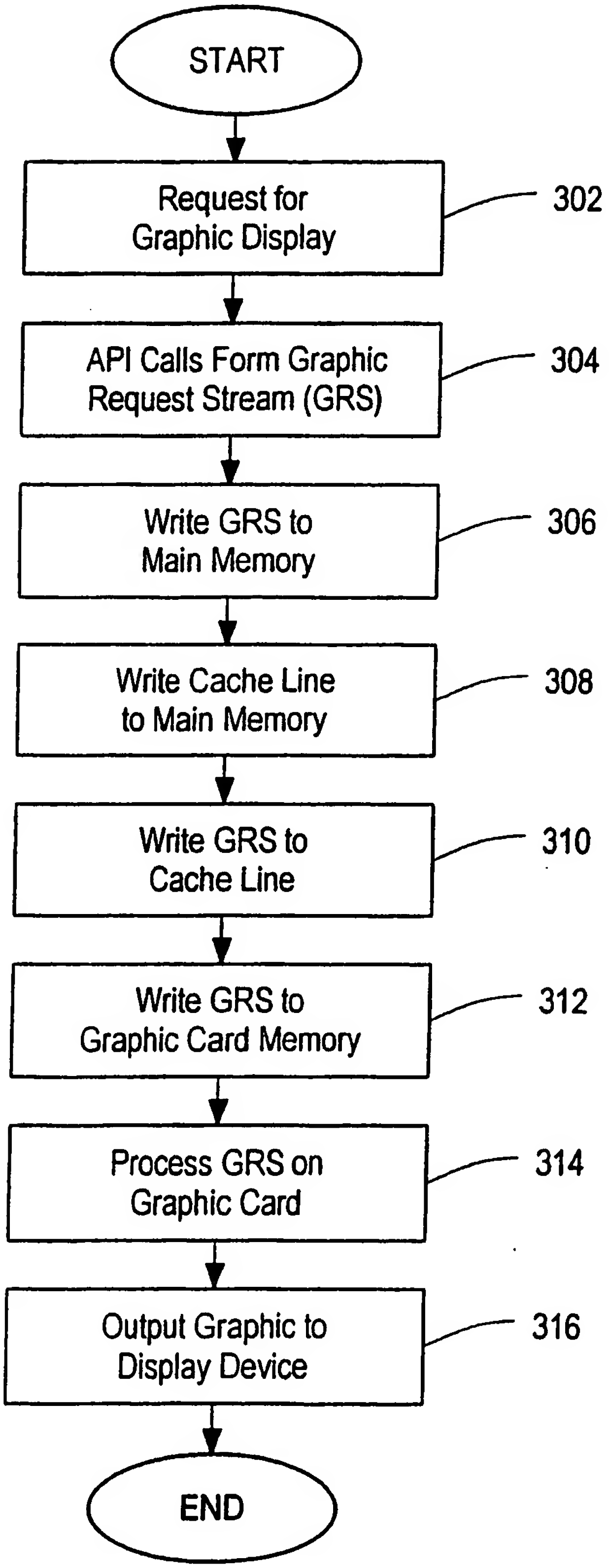

START
Request for Graphic Display
302
API Calls Form Graphic Request Stream (GRS)
304
Write GRS to Main Memory
306
Write Cache Line to Main Memory
308
Write GRS to Cache Line
310
Write GRS to Graphic Card Memory
312
Process GRS on Graphic Card
314
Output Graphic to Display Device
316
END

FIG. 2
Prior Art

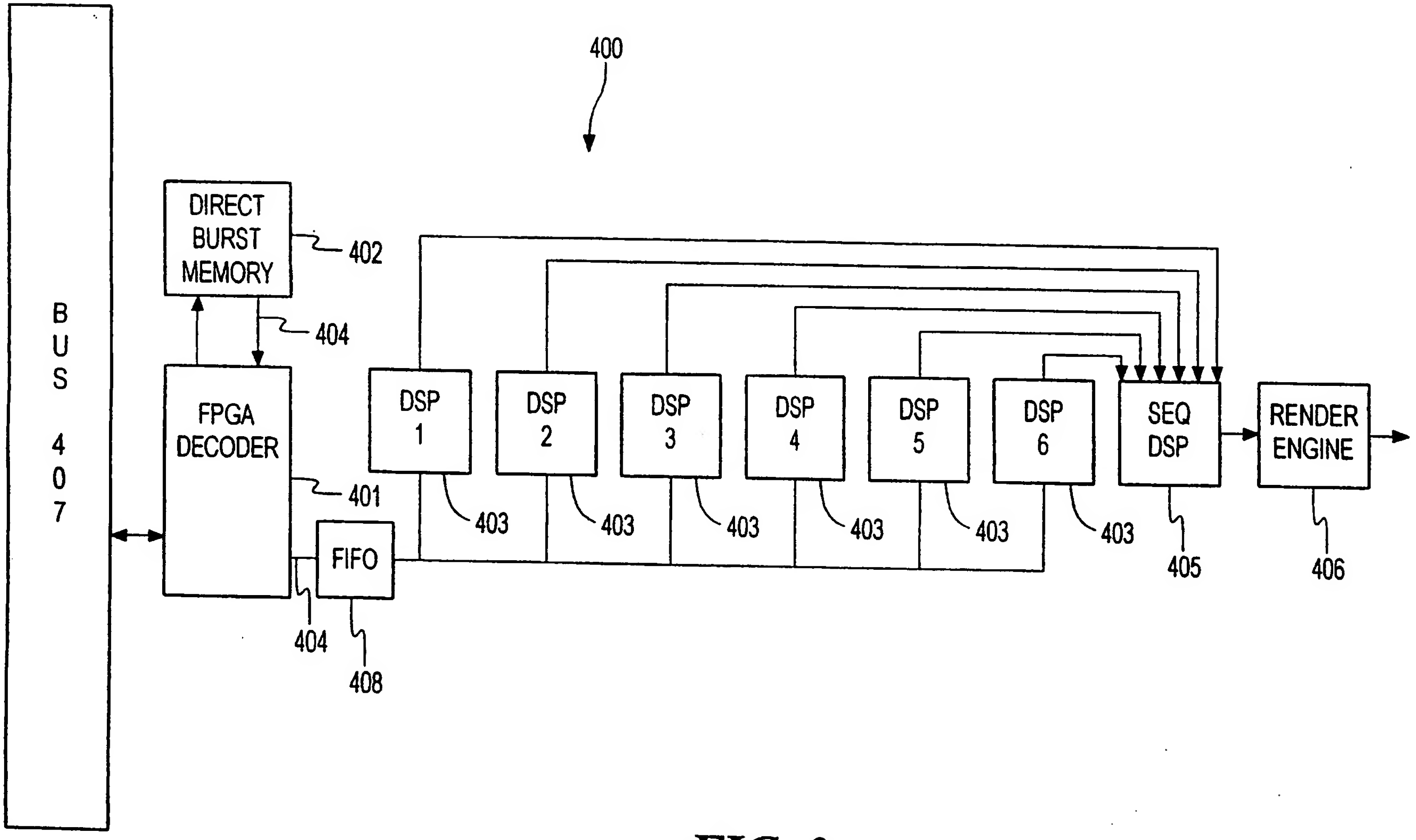

400
BUS 407
DIRECT BURST MEMORY
402
404
FPGA DECODER
401
FIFO
404
408
DSP 1
DSP 2
DSP 3
DSP 4
DSP 5
DSP 6
403
403
403
403
403
403
SEQ DSP
405
RENDER ENGINE
406

FIG. 3

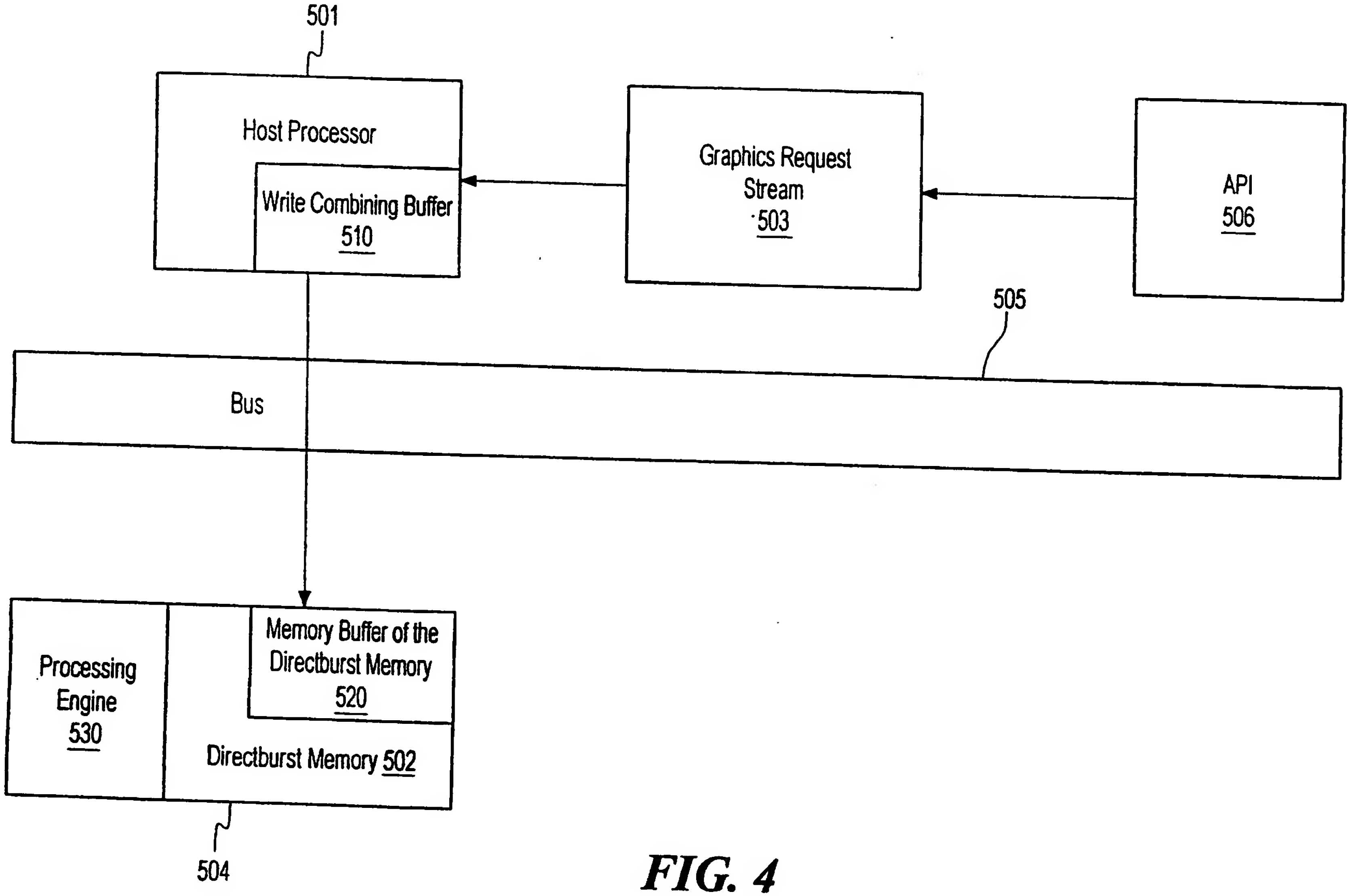

501
Host Processor
Write Combining Buffer
510
Graphics Request
Stream
503
API
506
505
Bus
Processing
Engine
530
Memory Buffer of the
Directburst Memory
520
Directburst Memory 502
504

FIG. 4

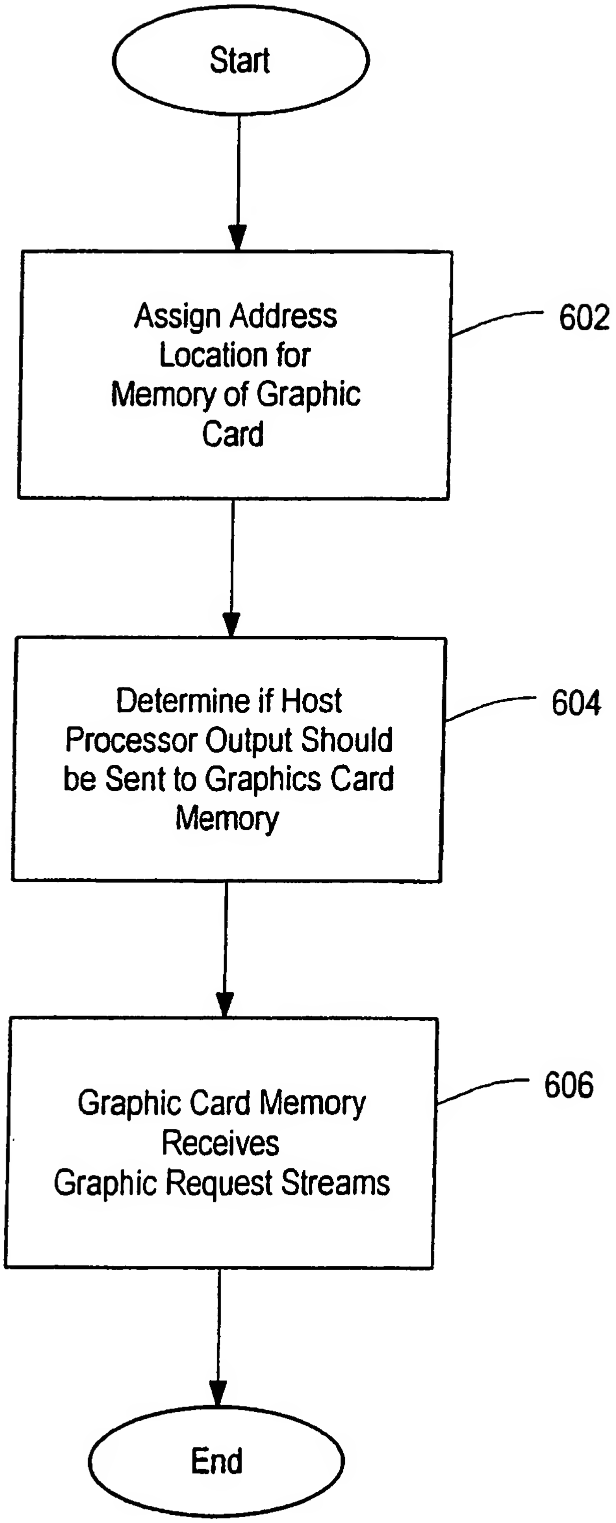

Start
Assign Address Location for Memory of Graphic Card
602
Determine if Host Processor Output Should be Sent to Graphics Card Memory
604
Graphic Card Memory Receives Graphic Request Streams
606
End

FIG. 5

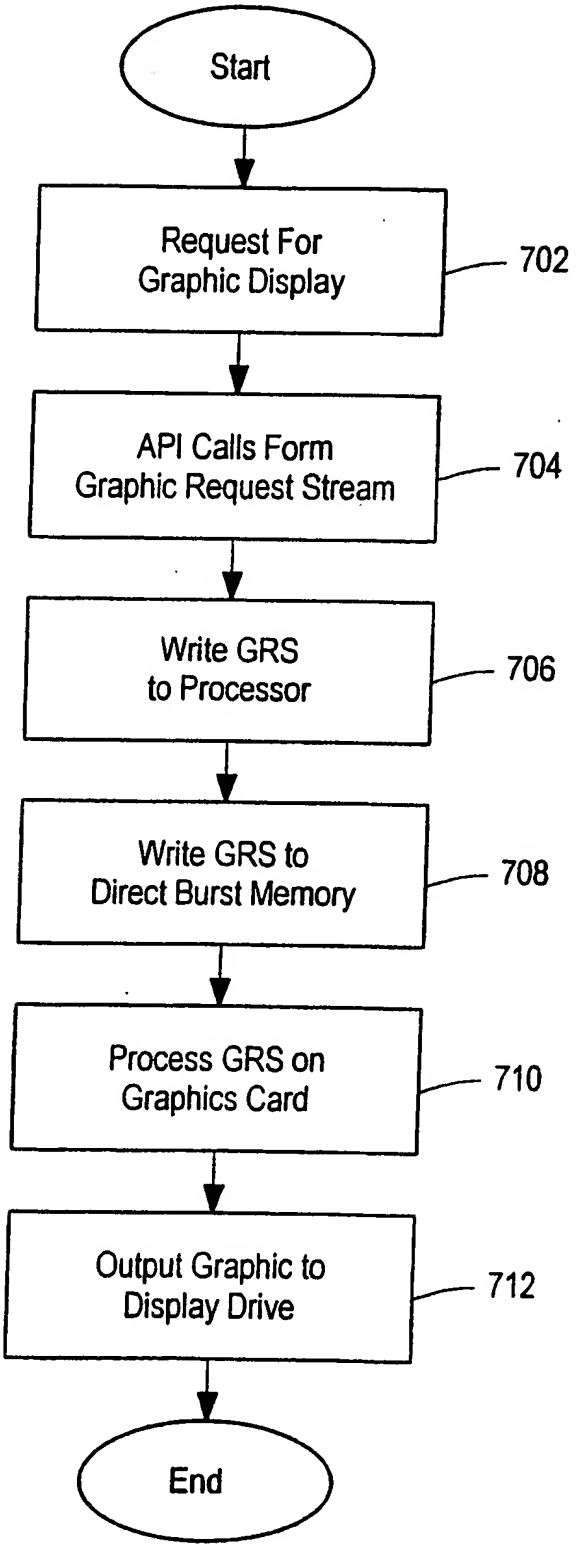

Start
Request For Graphic Display
702
API Calls Form Graphic Request Stream
704
Write GRS to Processor
706
Write GRS to Direct Burst Memory
708
Process GRS on Graphics Card
710
Output Graphic to Display Drive
712
End

FIG. 6

INTERNATIONAL SEARCH REPORT

Interr nal Application No
PCT/US 99/14889

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/38 G06F13/12 G06T1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 717 348 A (MOTOROLA INC) 19 June 1996 (1996-06-19)	1-3,5,6, 23-29
Y	column 1, line 28 -column 2, line 26	4,7-14, 16,18-22
A	column 2, line 52 -column 3, line 3 column 6, line 3 - line 56 column 12, line 11 -column 13, line 56 ---	4,8,12
Y	WO 93 04462 A (DIGITAL EQUIPMENT CORP) 4 March 1993 (1993-03-04) page 5, line 4 - line 28 page 7, line 9 - line 21 page 8, line 14 -page 9, line 11 page 13, line 22 -page 14, line 14 page 26, line 11 -page 29, line 37 ---	4,7-14, 16,18-22

-/--

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 29 October 1999	Date of mailing of the international search report 05/11/1999
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Daskalakis, T

INTERNATIONAL SEARCH REPORT

Intern nal Application No
PCT/US 99/14889

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 630 075 A (GLEW ANDREW F ET AL) 13 May 1997 (1997-05-13) the whole document ---	4,8,12
A	US 5 619 658 A (ROSENTHAL DAVID S H ET AL) 8 April 1997 (1997-04-08) column 4, line 27 -column 5, line 34 column 6, line 34 -column 9, line 23 ---	15-17
A	EP 0 483 970 A (IBM) 6 May 1992 (1992-05-06) column 2, line 55 -column 3, line 35 column 5, line 44 - line 55 column 7, line 53 -column 15, line 3 -----	1,7,14, 18,23,24

1

INTERNATIONAL SEARCH REPORT

.ormation on patent family members

Intern nal Application No
PCT/US 99/14889

Patent document cited in search report		Publication date	Patent family member(s)			Publication date
EP 0717348	A	19-06-1996	JP US	8339290 5889948	A A	24-12-1996 30-03-1999
WO 9304462	A	04-03-1993	US US AU	5313577 5321806 2553592	A A A	17-05-1994 14-06-1994 16-03-1993
US 5630075	A	13-05-1997	NONE			
US 5619658	A	08-04-1997	NONE			
EP 0483970	A	06-05-1992	WO HU IL JP PL SK US	9208192 65534 99744 4289955 166975 41393 5539896	A A,B A A B A A	14-05-1992 28-06-1994 26-05-1995 14-10-1992 31-07-1995 07-07-1993 23-07-1996